GROUP 3

Name	Work
Yashvardhan Toshniwal (CS22B088)	Userspace (File I/O subteam): ● Master wrapper integration ● Logger ● Hooked the system calls corresponding to file i/o. ● Created the testing env for testing all our wrappers. (Credits to Kritang for making the testing code) Kernel: ● Attempted some file io wrappers in kernel space ● Tried compiling the kernel image with the above wrappers Report: ● Weekly and final reports corresponding to the above parts.
Raadhes Chandaluru CS22B069	Kernel: ● Research on Compilation (4.17, 5.19, 6.07), booting and modification. Refer final report PAGE6. Kernel Syscalls/Wrappers: modified linux5.19, ● Disable custom syscall in root process of namespace ● disable_fork() & fork_if_not_disable() syscalls ● close_all_files() sys_call ● Tested with simple programs on booted modified linux kernel Userspace Wrappers: (Memory Subteam) ● File memory leak prevention wrappers (tracked_open, tracked_close, close_all_files). Modified tracked_<call> to open and close with hooks. (Credits: Aditya Srivastava for hook procedure) ● Shared memory loggers (shm<> wrappers) Report: ● Final Report: Kernel Section. ● Week 2 & 3: Respective wrappers.
Kritang Kothari CS22B012	Wrappers (File I/O): ● Safe_open ● Buffer Testing: ● Verified the functionality and correctness of all file I/O wrappers by executing the master wrapper's program on various test cases. Hooks: ● Learned about hooks and how they are used to create wrappers around system calls. (Credits to Yashvardhan for implementation)

	Report: ● Weekly and final reports corresponding to the above parts.
Daksh Sehra	Wrappers (File I/O): ● Safe_read ● Rate limiter ● Control Permission Hooks: ● Read about usage of hooks. Testing: ● Tested the functionality of the mentioned wrappers by executing each one of them separately. Report: ● Made reports corresponding to the mentioned wrappers.
Aditya Jain CS22B065	**Wrappers**: ● **Wait For All children.** ● **Process Monitoring and Logging.** ● **Priority Enforcement Wrapper** (run the executable using sudo) —> in collaboration with Shreyanshu Gurjar. ● **Process Pool Manager.** **Hooks**: ● Implemented Wait for All children and Process Monitoring and Logging wrappers using dlsym provided by the **dynamic linking library (`libdl`)** to dynamically resolve symbols (like functions and variables) at runtime from shared libraries and LD_PRELOAD to load our custom shared object before any other object. **Report**: ● Week 1, week 2, week 3, presentation and the final report corresponding to the wrappers mentioned above. **Testing**: ● Checked the correctness of the above four wrappers by running them against a sample test case.
Aditya Srivastava CS22B066	Wrappers (Implemented with hooks) - ● Deadlock detection wrapper ● Process Cloaking Wrapper ● Comprehensive Inter-Process Communication Logging Wrapper Hooks - ● Researched and implemented hooks as a method of implementing wrappers that can wrap existing syscalls with the exact same function primitive in user space

	using shared object libraries, dlsym (system calls table), and Linux feature LD_PRELOAD which loads the shared object library first. Report - ● Proposal, week1, week2, week3 and final report. Testing - ● Tested all above wrappers against comprehensive sample test cases.
Dev Mehta CS22B007	Wrappers - ● Safe munmap wrapper ● Debug malloc wrapper ● Debug free wrapper ● Heap corruption wrapper Hooks - ● Research about hooks, how to implement syscall wrappers using dlsym (to get address of a symbol defined within an object), LD_PRELOAD to load our custom shared object before any other objects (including libc.so). ● Implemented memory related syscalls with hooks. Testing - ● Verified the functionality and correctness of all memory wrappers by executing the programs on various test cases. Report - ● Proposal, Week1, Week2, Week3, presentation and the final report corresponding to the wrappers as mentioned above.
Shreyanshu Gurjar CS22B084	Wrappers - ● Comprehensive IPC Logging Wrapper ● Priority Enforcement Wrapper - in collaboration with Aditya Jain Report - ● Proposal, Week2 and Week3 report corresponding to the wrappers mentioned above. Testing - ● Verified the functionality and correctness of some of the process wrappers by executing the programs on various test cases. Hooks - ● Research about user space hooking to implement process wrappers using dlsym and LD_PRELOAD.
Harsh Vardhan Daga cs22b075	Wrappers - ● Custom Waitpid ● Zombie Process Logger (waitpid) ● Zombie Process Logger (kill)

	Hooks - ● Learned and read about hooks and their user level implementation. Also implemented hooks using dlsym(which is used to get the address of a particular syscall)and LD_PRELOAD which is used to load the custom shared object. Testing - ● Verified the functionality and correctness of the above process wrappers by executing the program on various test cases. Report - ● Week2, week3 and the final report corresponding to the wrappers mentioned above.
Mith R Jain	Wrappers: ● safe_mmap ● brk ● memory_pool_call Report: ● Week 2, Week 3 and the Final Report corresponding to the wrappers mentioned above Hooks: ● Read up about hooks and added the hook part of the wrapper for the memory wrappers using dlsym
Rohan Bagati CS22B082	Wrappers: ● aligned_mmap ● aligned_munmap Report: ● Week 2, Week 3 and the Final Report corresponding to the wrappers mentioned above. Testing - ● Verified the functionality and correctness of the above memory wrappers by executing the program. Hooks: ● Read up about hooks and collaborated with my team in implementing them on these memory wrappers.